# BAB III

# METODOLOGI DAN PERANCANGAN APLIKASI

## 3.1 Metodologi Penelitian

Adapun tahap-tahap metodologi penelitian aplikasi *keyboard* pemblokir kata pornografi adalah sebagai berikut.

1. Studi Kelayakan

Studi Kelayakan ini dilakukan untuk mengetahui apakah memang benar anggota NoFap Indonesia kecanduan masturbasi dan pornografi serta mengetahui apakah aplikasi yang akan dibuat memang diperlukan atau tidak. Pengukuran ini dilakukan dengan penyebaran kuesioner dengan minimal 30 (Sugiyono, 2012).

2. Telaah Literatur

Mengumpulkan dan mempelajari literatur yang relevan untuk mencapai hasil penelitian yang sesuai dengan teori yang ada.

3. Analisis Sistem

Menganalisa kebutuhan berdasarkan permasalahan yang terjadi, serta mengumpulkan data yang diperlukan dalam pembuatan sistem.

4. Desain Sistem

Membuat antarmuka yang menarik, serta keseluruhan *database* terkait dengan hasil aplikasi. Hal ini dapat dilakukan dengan membuat diagram dengan menggunakan UML (*Unified Modeling Languange*). UML dipakai karena biasa digunakan sebagai pemodelan berorientasi objek. UML sendiri terdiri dari *use case diagram, activity diagram, sequence diagram,* dan *class diagram*.

5. Pemrograman Sistem

Melakukan pemrograman sesuai dengan desain sistem yang telah selesai dirancang.

6. Pengujian Sistem

Tahap pengujian sistem ini dlakukan saat pemrograman sistem telah selesai dilakukan. Pengujian *usability* akan dilakukan dengan *System Usability Scale* (SUS). Pengujian kecepatan akan dilakukan dengan mencatat waktu yang dibutuhkan dalam proses pencarian kata dalam algoritma.

7. Evaluasi

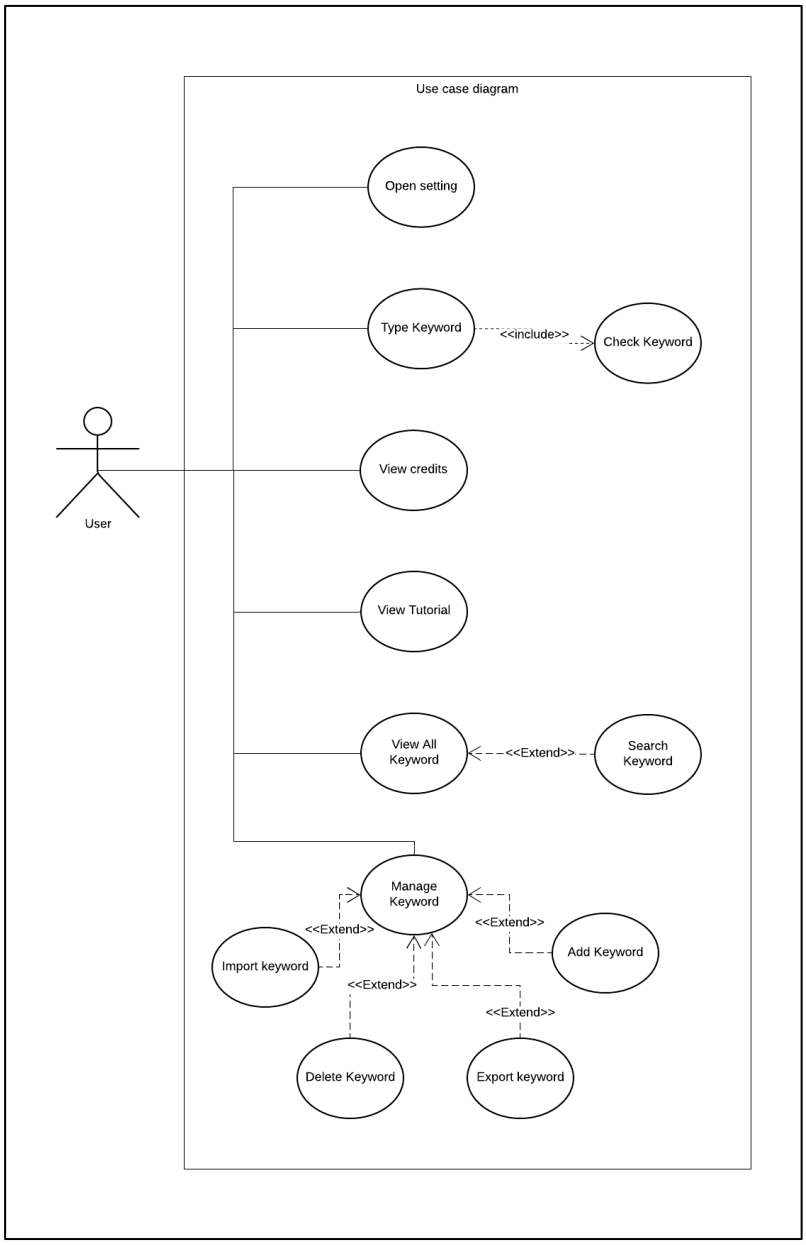
Setelah pengujian sistem dilakukan, evaluasi diambil berdasarkan hasil dari *System Usability Scale* dan hasil pengujian kecepatan proses pencarian kata.

## 3.2 Perancangan Aplikasi

Berikut rancangan aplikasi yang telah dibuat menggunakan UML yang dijabarkan menjadi *use case diagram, activity diagram, sequence diagram,* dan *class diagram*.

### 3.2.1 Use Case Diagram

Gambar 3.1 menunjukan *Use case diagram* pada aplikasi ini. Pada Gambar 3.1 tersebut dapat dilihat bahwa terdapat 1 buah aktor yaitu *user* yang merupakan pemakai dari aplikasi. Pada *use case* tersebut *user* terhubung dengan 10 *use case* yaitu *add keyword, search keyword, view all keyword keyword, delete keyword, type keyword, tutorial, import keyword, export keyword, open setting, dan credits.*




Gambar 3.1 *Use Case Diagram*

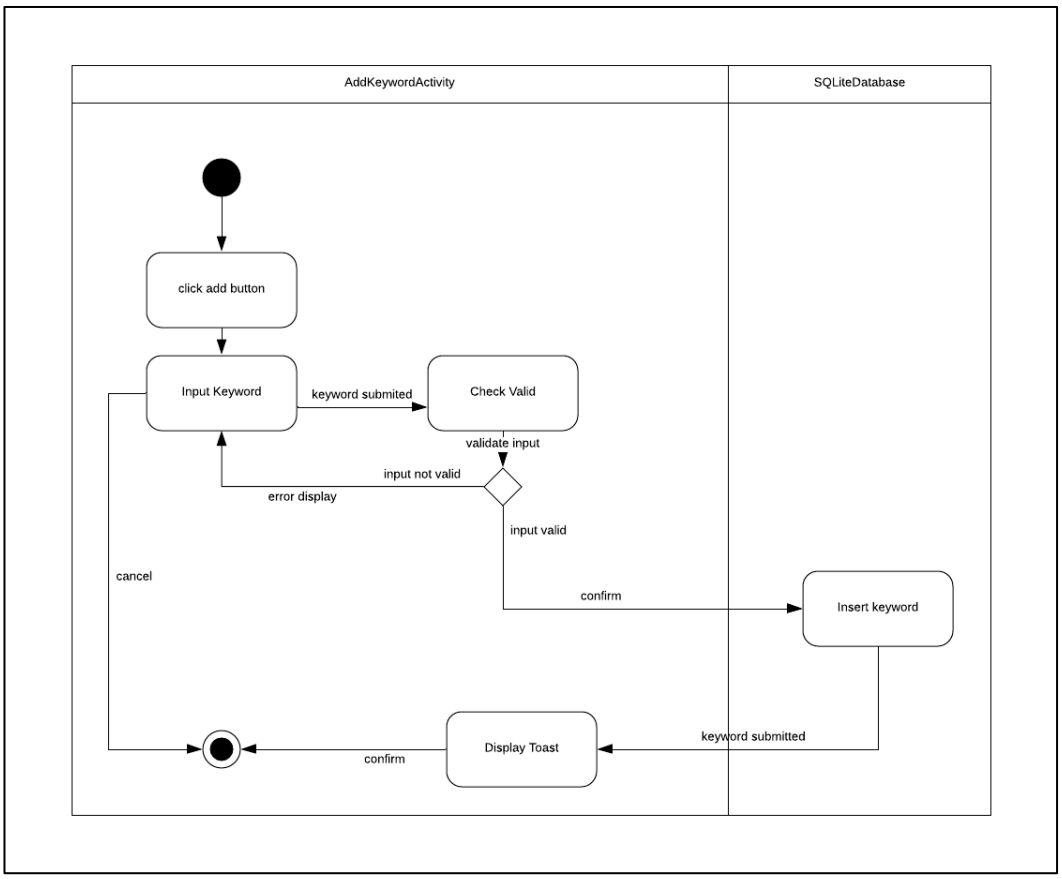
*Use case check keyword* ter-*include* di *use case type keyword* karena pada saat *user* melakukan *typing* kalimat selayaknya mengetik biasa di *keyboard* yang terbuat dari aplikasi ini, secara bersamaan *check keyword* juga dijalankan. *Use case add keyword, delete keyword, import keyword, dan export keyword* merupakan *extended use case* dari *use case manage keyword* karena *user* dapat melakukan hal tersebut namun hal tersebut tidak harus dilakukan oleh *user*. Begitu juga dengan

*search keyword* yang juga *extend* dari *view all keyword* karena *user* dapat melakukan hal tersebut namun hal tersebut tidak harus dilakukan oleh *user*

### 3.2.2 Activity Diagram

Pada Gambar 3.1 juga terdapat beberapa *activity diagram* untuk *use case* yang telah dibuat.

A. Activity Diagram Add Keyword

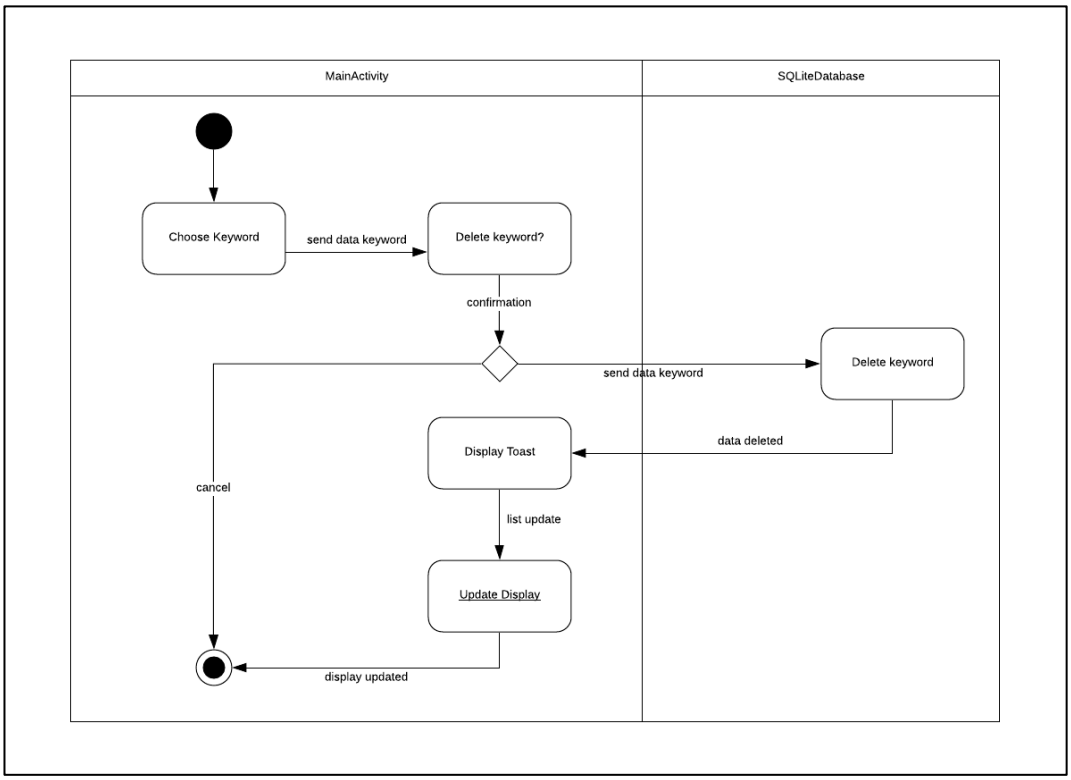



Gambar 3.2 *Activity Diagram Add Keyword*

*Activity diagram add keyword* dapat dilihat pada Gambar 3.2. Untuk melakukan *add keyword*, pertama *user* harus menekan add button lalu mengisi data *keyword*. Data kemudian di-*check* oleh aplikasi, yang di mana dilakukan pengecekan apakah *input user* tersebut *valid* (berisi) atau tidak (kosong). Jika *input valid* maka akan

diteruskan ke *database* lalu diberikan pesan ke *user* bahwa *keyword* berhasil dimasukan. Sebaliknya jika *input keyword* tidak *valid* maka akan ditampilkan pesan *error* kepada *user*.

B. Activity Diagram Delete Keyword

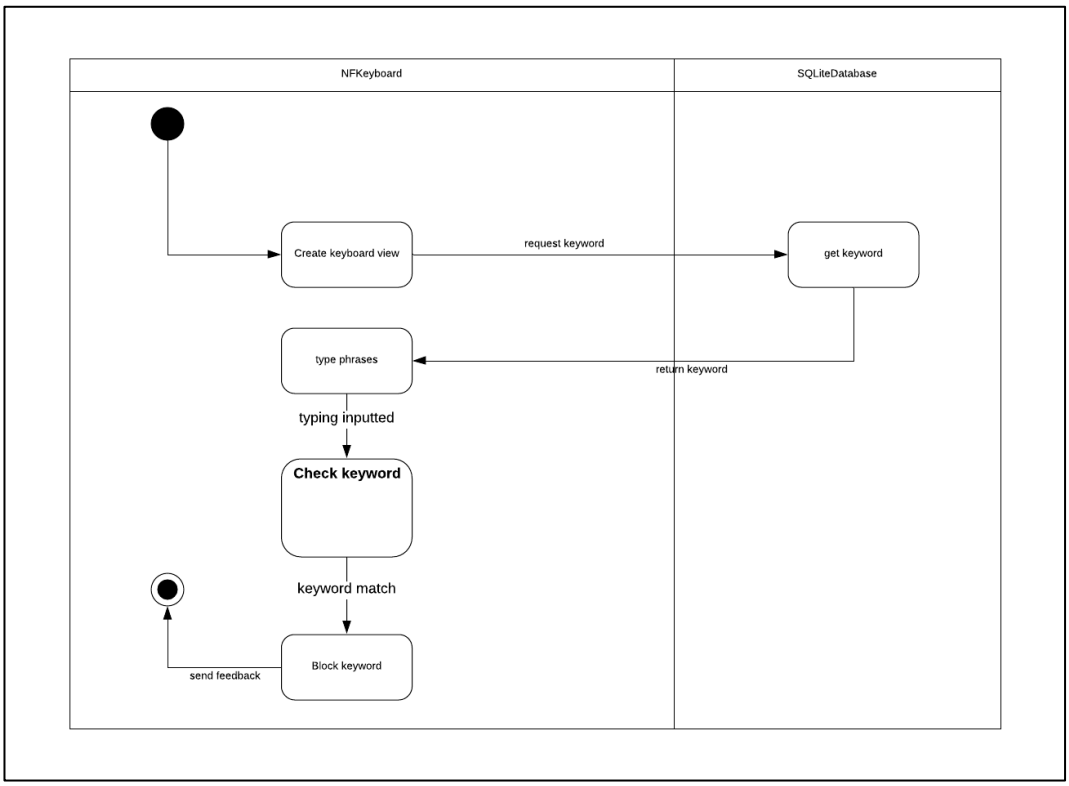



Gambar 3.3 *Activity Diagram Delete Keyword*

*Activity diagram delete keyword* dapat dilihat pada Gambar 3.3. Untuk melakukan *delete keyword*, *user* harus memilih data mana yang akan dihapus terlebih dahulu. Selanjutnya ketika *user* memilih data *keyword*, maka akan muncul konfirmasi apakah *user* yakin ingin menghapus atau tidak. Jika ingin dihapus, data *keyword* yang tadi secara bersamaan dilempar saat muncul kotak *dialog* juga diambil oleh aplikasi lalu dilanjutkan ke *database* untuk dipilih dan dihapus. Setelah data dihapus *database*, lalu diberikan pesan kepada *user* bahwa data telah

berhasil dihapus. *Display list data keyword* akan *diupdate* oleh aplikasi untuk mengetahui data yang tersisa.

C. Activity Diagram Type Keyword

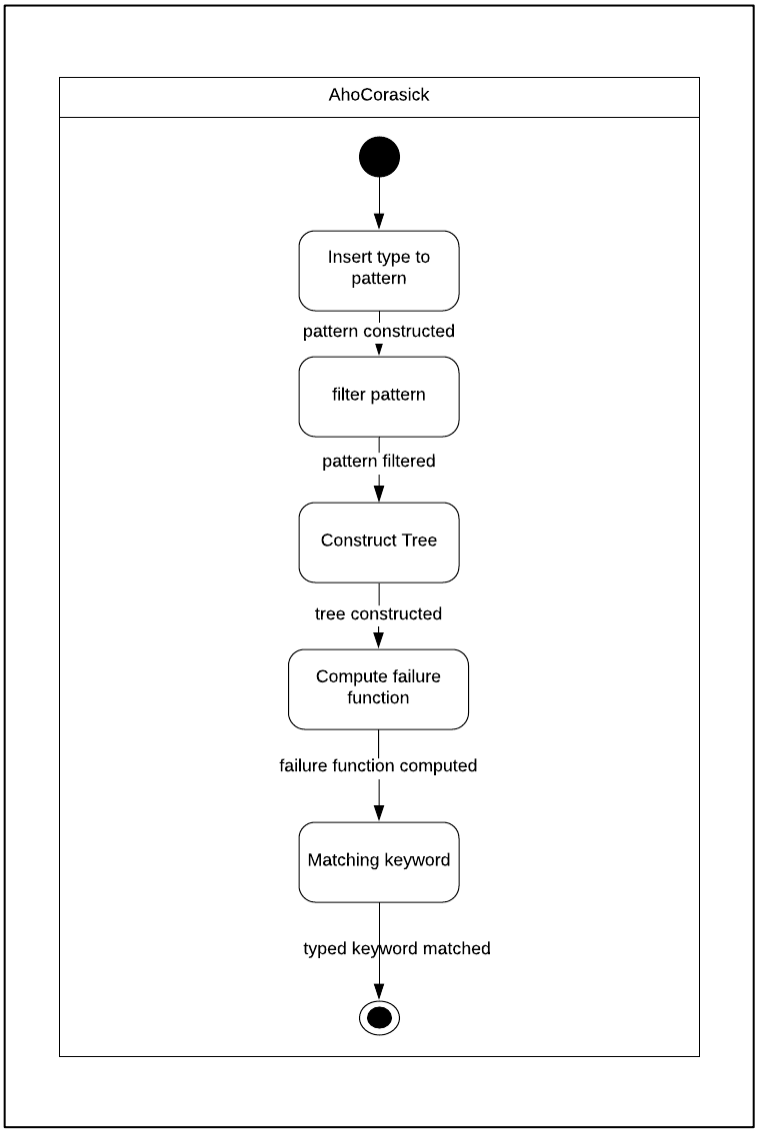



Gambar 3.4 *Activity Diagram Type Keyword*

Gambar 3.4 menunjukan *diagram* dari *activity diagram type keyword*. *Activity diagram type keyword* dijalankan pada saat *user* melakukan pengetikan pada *keyboard* bawaan aplikasi yang telah dibuat. Pada saat *user* meng-*click input box*, secara bersamaan aplikasi juga membuat *keyboard view* untuk ditampilkan dan me-*request* data *keyword* di *database* untuk melakukan pengecekan *keyword*. Setelah data didapat, maka akan langsung diteruskan ke bagian *check keyword*.

*Check keyword* akan membandingkan kalimat yang diketik dan *keyword* yang ada di *database* dengan algoritma *Aho-Corasick*. Jika hasilnya sesuai maka kata tersebut akan di-*block* dan dihilangkan oleh aplikasi.

D. Activity Diagram Check Keyword



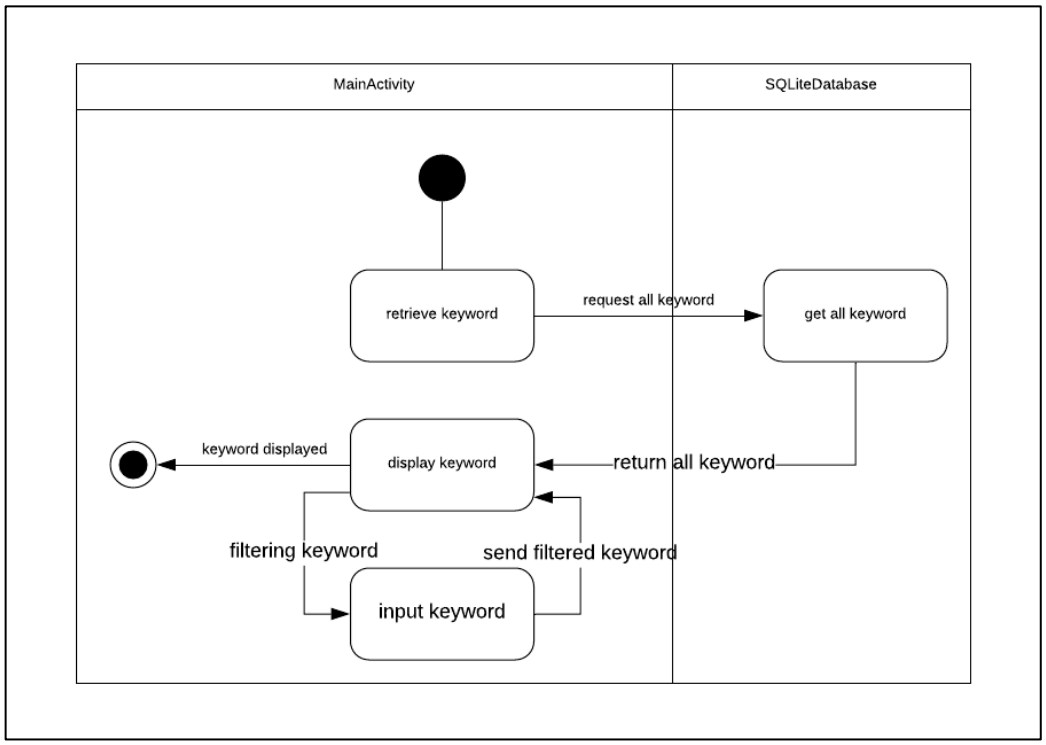

Gambar 3.5 *Activity Diagram Check Keyword*

Gambar 3.5 menunjukan *diagram* dari *activity diagram check keyword*. Pertama kali proses yang dijalankan adalah memasukan *keyword* yang ada di *database* ke dalam *pattern*. Setelah itu, akan di-filter keyword tadi agar di tree tidak double jikalau memang ada keyword di database yang persis sama. Lalu dibuatlah *tree pattern* tersebut. Setelah itu akan dilakukan pengecekan, jika karakter yang diketik tidak sesuai dengan karakter di dalam *tree*, akan dilakukan proses *failure function* yang mana akan mengecek ulang karakter dari *root tree* ke *state* lain. Jika

sesuai, maka akan dimasukan ke dalam *variable* baru & dilakukan persamaan dengan *keyword* di dalam *database* yang telah diambil tadi. Jika sesuai, akan di-*return* dan di-*block* oleh aplikasi.

D. Activity Diagram View All Keyword

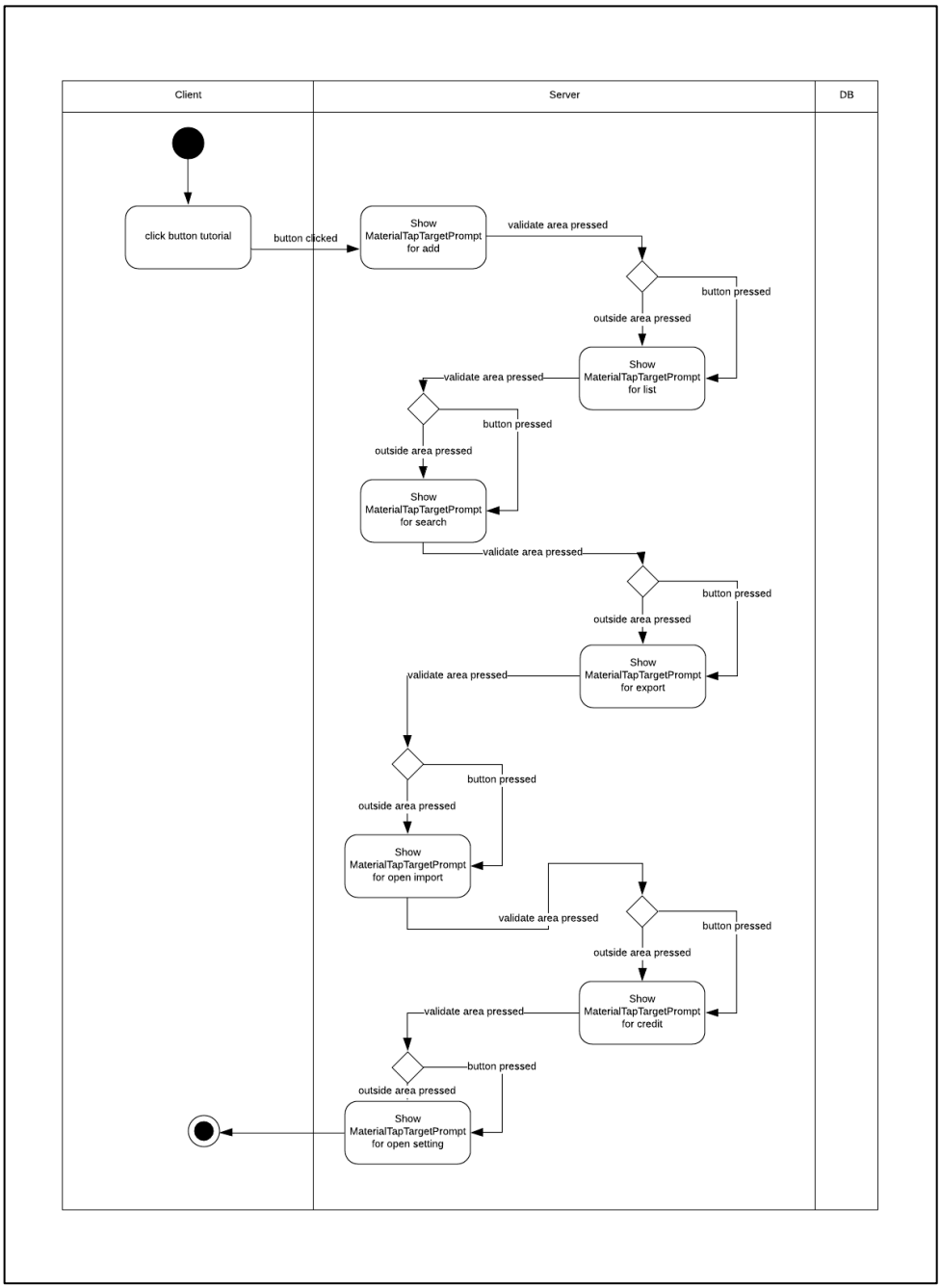
*Activity diagram view all keyword* dapat dilihat pada Gambar 3.6. *Untuk* melakukan *view all keyword,* aplikasi pertama kali akan data *keyword* dari *database.* Lalu seluruh data *keyword* diambil, data tersebut akan di-*set* dan di-*display* ke *user*. Setelah itu *recycler view* tersebut akan ditampilkan kepada *user*. User juga dapat melakukan *filtering keyword* atau *search*. *Search* dilakukan berdasarkan *keyword* yang dimasukkan oleh *user* ke dalam *search bar*. Setelah *user* memasukkan *keyword* yang ingin dicari, maka *keyword* akan langsung di-*filter* dan *keyword* hasil *filtering* pun akan ditampilkan.




Gambar 3.6 *Activity Diagram View All Keyword*

F. Activity Diagram Tutorial

*Activity diagram tutorial* dapat dilihat pada Gambar 3.7. Untuk melakukan *tutorial*, pertama kali *user* harus menekan *tutorial button*. Setelah itu aplikasi akan menampilkan *material tap target prompt* yang di mana merupakan *library*.

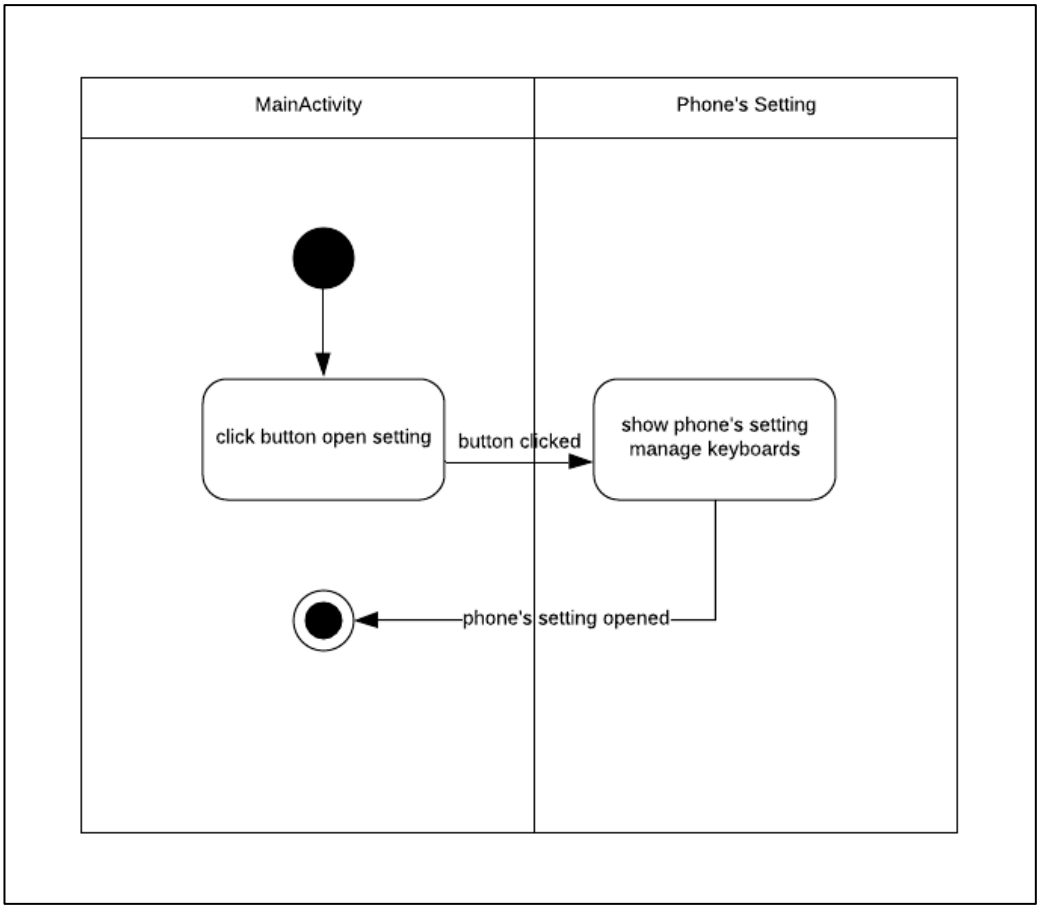



Gambar 3.7 *Activity Diagram Tutorial*

Alur penampilan *material tap target prompt* dimulai dari *material tap target prompt add keyword*, setelah itu jika *button* atau *area* diluar *button* ditekan maka akan lanjut ke *material tap target prompt list keyword*. Alur akan berjalan seperti itu sampai *user* menekan *material tap target prompt* untuk *button seting*. Setelah *material tap target prompt setting* ditekan, lalu *tutorial* pun akan selesai.

G. Activity Diagram Open Setting

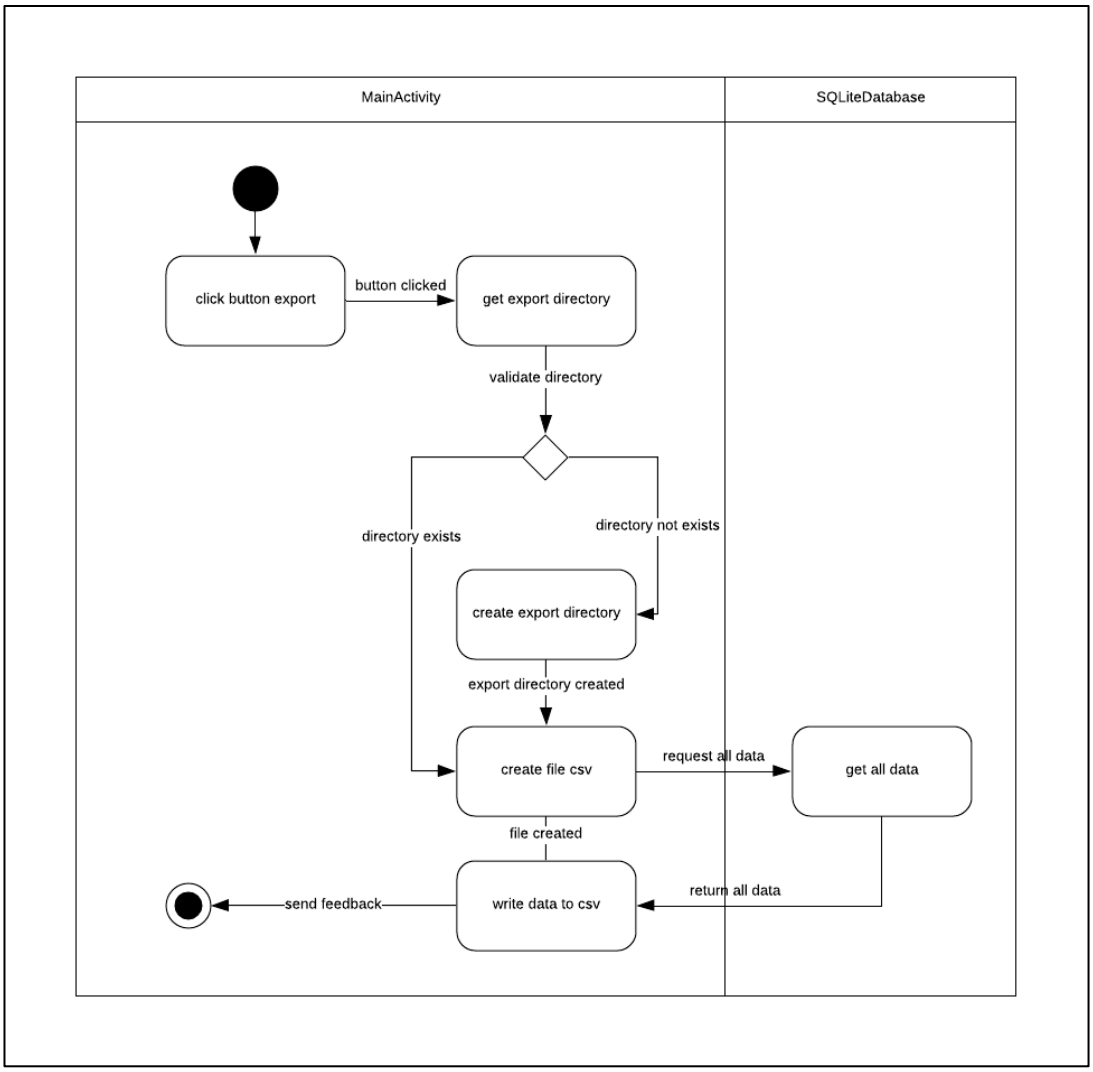
*Activity diagram Open Setting* dapat dilihat pada Gambar 3.8. Untuk membuka *setting*, pertama kali *user* harus menekan *open setting button*. Setelah itu aplikasi akan langsung menampilkan *setting area* di *OS handphone* masing-masing di mana *user* dapat meng-*enable* NFKeyword.




Gambar 3.8 *Activity Diagram Open Setting*

H. Activity Diagram Export Keyword

*Activity diagram Export Keyword* dapat dilihat pada Gambar 3.9. Untuk melakukan export keyword, pertama kali *user* harus menekan *export button*. Setelah itu aplikasi akan langsung mengambil *export directory* yang di mana jika belum dibuat sebelumnya, akan dibuat terlebih dahulu.



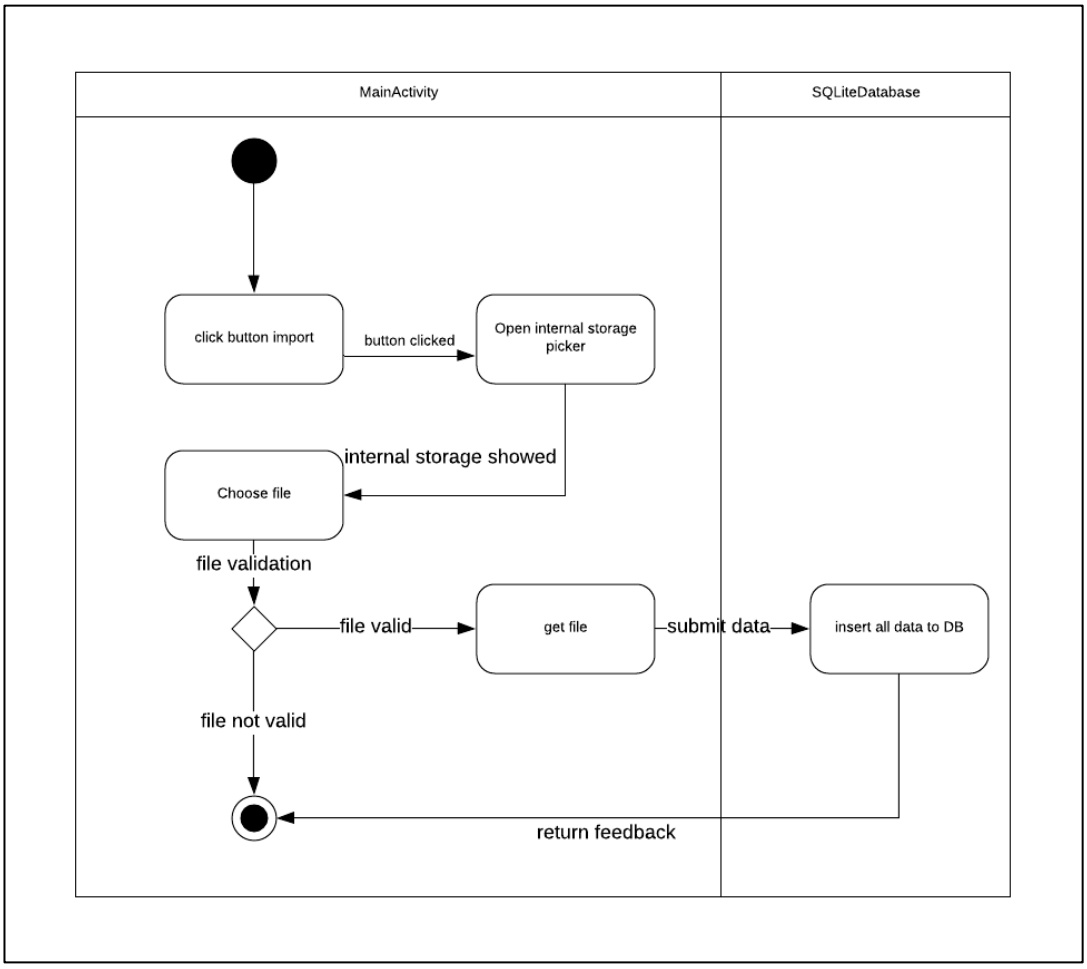

Gambar 3.9 *Activity Diagram Export Keyword*

Setelah itu aplikasi akan membuat *file* CSV. Lalu aplikasi akan mengambil dan menyimpan seluruh data yang ada di *database* ke *file* CSV tersebut, jika sudah

selesai aplikasi akan langsung memberikan *feedback* kepada *user* bahwa proses *export* telah selesai dilakukan.

I. Activity Diagram Import Keyword

*Activity diagram Import Keyword* dapat dilihat pada Gambar 3.10. Untuk melakukan import keyword, pertama kali *user* harus menekan *import button*. Setelah itu aplikasi akan membuka *internal storage* dengan *library internal storage picker*.



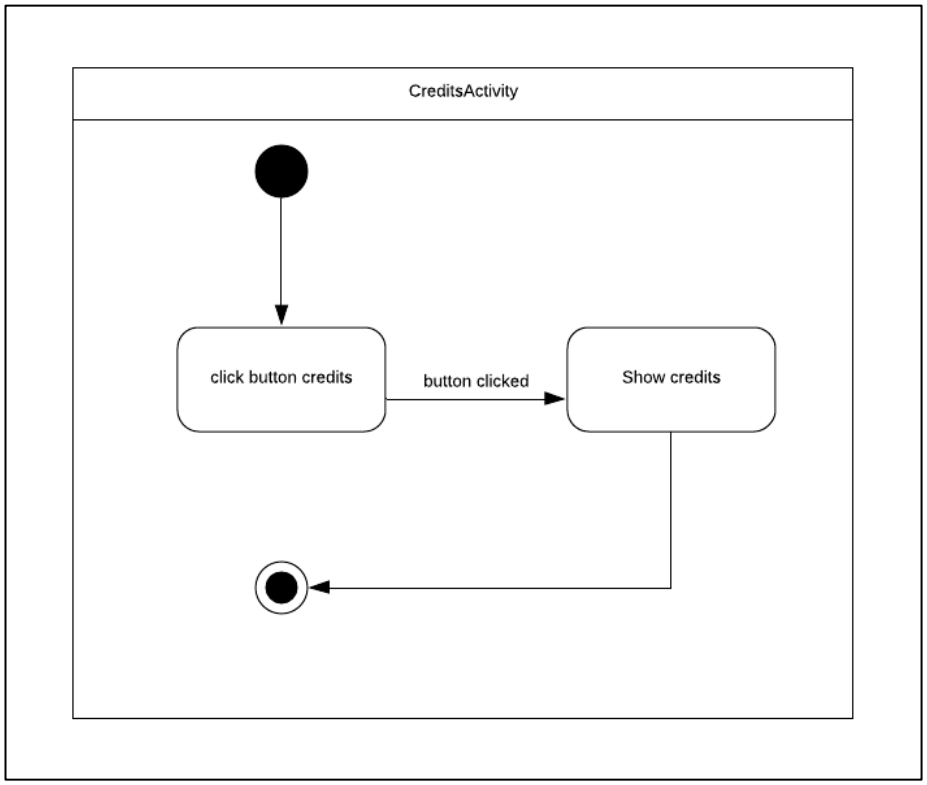

Gambar 3.10 *Activity Diagram Import Keyword*

*File* selanjutnya akan dipilih oleh *user* dan divalidasi apakah merupakan *file* yang bernama *keyword.csv* atau bukan. Jika iya, aplikasi akan mengambil *file* tersebut, lalu membuka & membaca seluruh isi *file*. Setelah itu isi *file* akan di-*write*

ke dalam *database*. Jika sudah selesai, aplikasi akan memberikan *feedback* pertanda bahwa *import* telah selesai dilakukan.

J. Activity Diagram Credits

*Activity diagram credits* dapat dilihat pada Gambar 3.11. Untuk melakukan *credits*, pertama kali *user* harus menekan tombol *credits button*. Setelah itu akan langsung ditampilkan halaman *credits* kepada *user*.



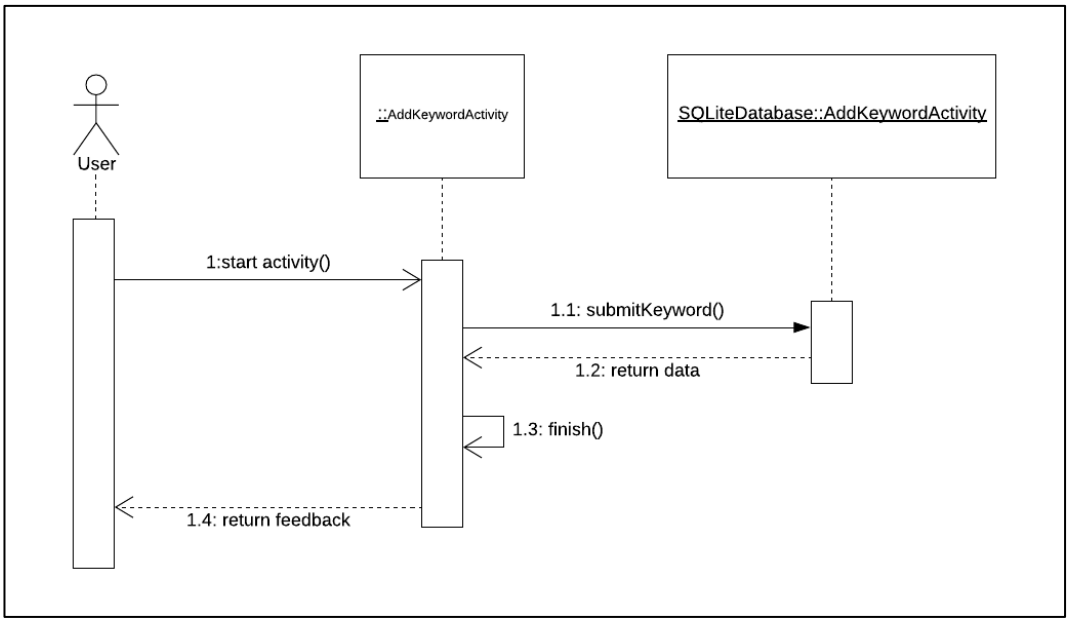

Gambar 3.11 *Activity Diagram View Credits*

### 3.2.3 Sequence Diagram

Terdapat 10 sequence diagram yang dirancang untuk aplikasi ini yaitu *sequence diagram add keyword, check keyword, view all keyword keyword, delete keyword, type keyword, tutorial, import keyword, export keyword, open setting, dan view credits.*

A. Sequence Diagram Add Keyword

*Sequence* diagram untuk *add keyword* dapat dilihat pada Gambar 3.12. *Activity add keyword* berjalan pada saat *user* meng-*click button add keyword* pada *main activity*. Lalu akan di-*return* kepada *user* berupa tampilan *add keyword activity*. Setelah itu *user* harus menekan tombol *add* yang telah tersedia untuk menjalankan fungsi submitKeyword() yang akan memasukan *keyword* ke dalam *database.* Secara bersamaan juga ada *function* finish() yang berfungsi untuk mengakhiri *activity* yang sedang berjalan sebelumnya. Lalu diteruskan dengan return *feedback* bahwa *add keyword* berhasil.

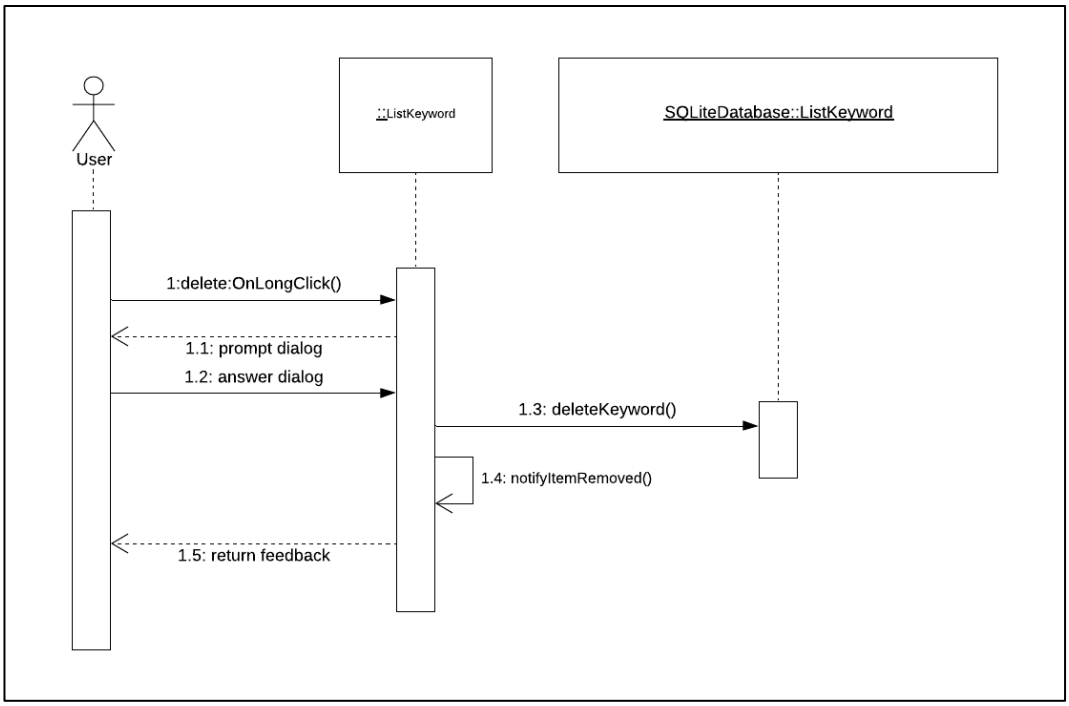



Gambar 3.12 *Sequence Diagram Add Keyword*

B. Sequence Diagram Delete Keyword

*Sequence* diagram untuk *delete keyword* dapat dilihat pada Gambar 3.13. Pada diagram ini, yang pertama kali dilakukan *user* adalah meng-*long click* tiap data *keyword* yang ingin dihapus. Karena setiap data *keyword* yang ada di *database* telah ditarik dan ditampilkan di *main activity*. Setelah data di-*long click*, maka akan muncul kotak dialog untuk menanyakan kepada *user* apakah yakin ingin

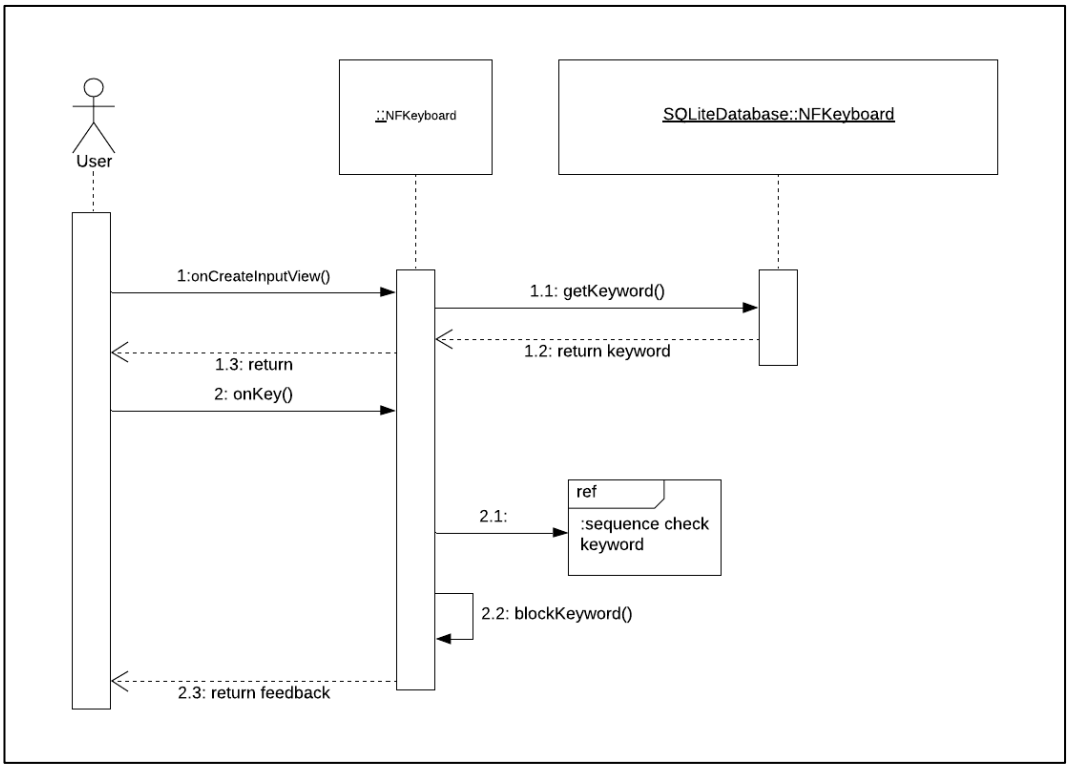
menghapus data. Setelah *feedback* hasil dialog dikembalikan *user*, dilakukan *function* deleteKeyword() di *database*. *Function* NotifyItemRemoved() dipakai setelahnya untuk melakukan pemberitahuan bahwa aplikasi harus meng-*update* data karena ada data yang telah terhapus. Setelah *function* NotifyItemRemoved() dijalankan, diteruskan *feedback* ke *user* bahwa data telah berhasil dihapus.




Gambar 3.13 *Sequence Diagram Delete Keyword*

C. Sequence Diagram Type Keyword

*Sequence* diagram untuk *type keyword* dapat dilihat pada Gambar 3.14. *Sequence* diagram *type keyword* dijalankan pada saat *user* melakukan pengetikan pada *keyboard* bawaan aplikasi yang telah dibuat.



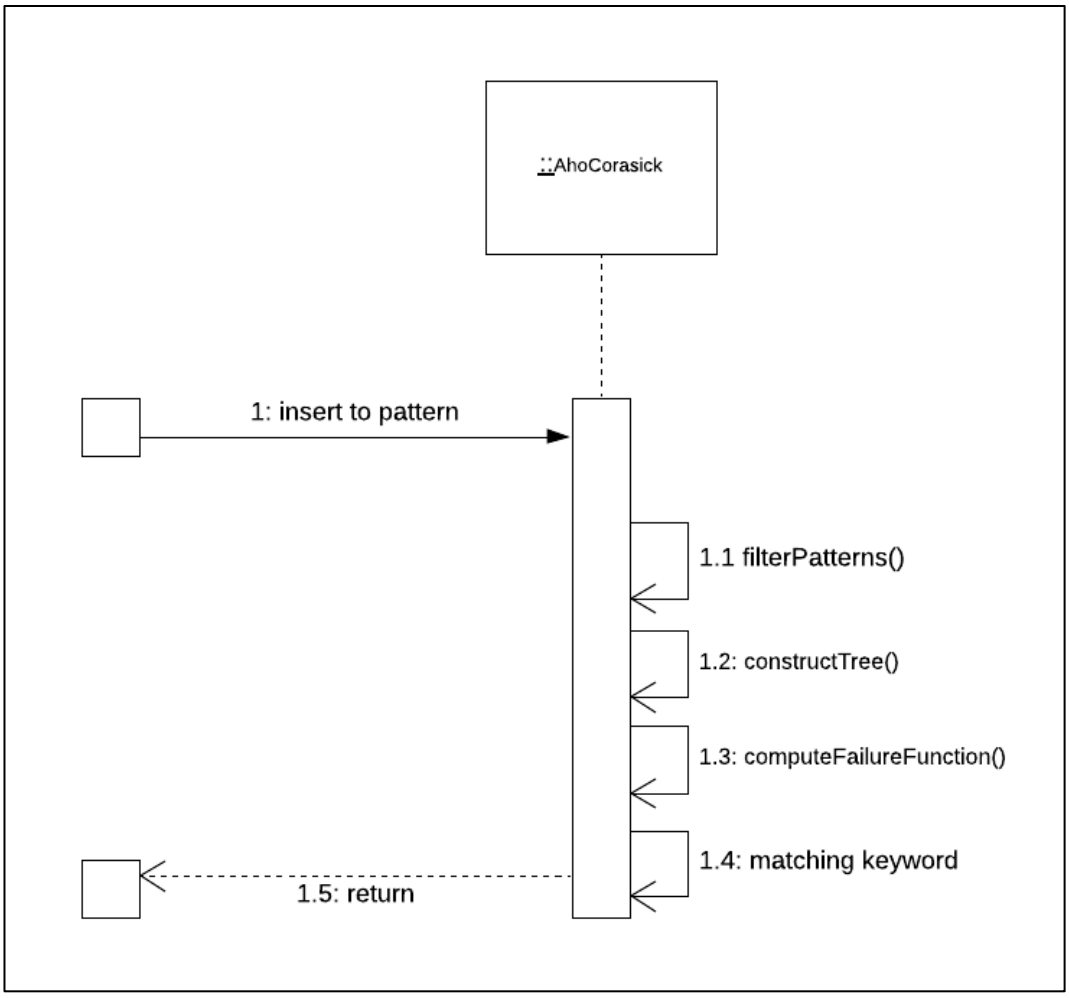

Gambar 3.14 *Sequence Diagram Type Keyword*

Pada saat *user* meng-*click edit text* di *platform* apapun, secara bersamaan aplikasi juga melakukan *function* onCreateInputView() untuk membuat tampilan *keyboard* bawaan aplikasi. *Setelah keyboard view* terbuat, diambil *keyword* dari *database* dengan function getKeyword() dan langsung diteruskan ke aplikasi. *Function* onKey() adalah *function* yang digunakan pada saat *button keyboard* diketik, setelah itu akan dilakukan *check keyword*.

Di dalam *check keyword*, jika hasilnya sesuai maka dengan menggunakan *function* blockKeyword() kata tersebut akan di-*block* dan dihilangkan oleh aplikasi. Lalu *feedback* akan diberikan kepada *user* yang berupa tanda bahwa kata telah di-*block*.

D. Sequence Diagram Check Keyword

*Sequence* diagram untuk *check keyword* dapat dilihat pada Gambar 3.15. *Sequence* diagram *check keyword* dijalankan setelah user mengetik dan system telah melakukan pembersihan kalimat yang diketik pada *sequence diagram type keyword*.



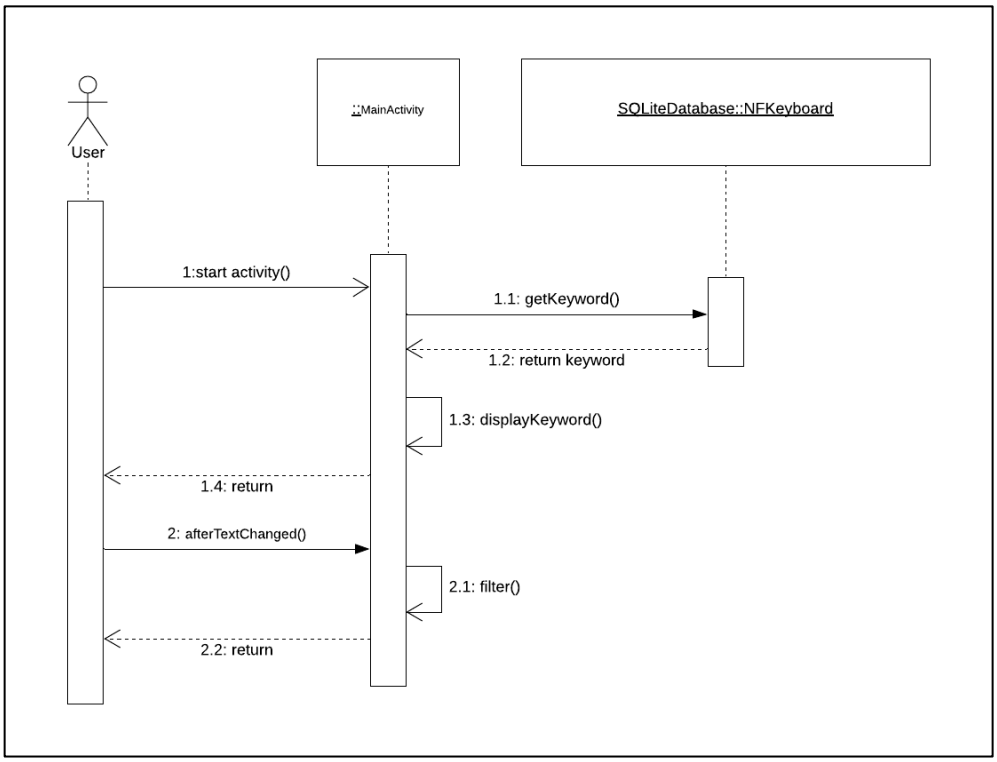

Gambar 3.15 *Sequence Diagram Check Keyword*

Pada *sequence diagram check keyword* ini, *keyword* yang ada di *database* akan diambil, diteruskan dan dimasukkan ke dalam *patterns*. Setelah *patterns* terbentuk dan terisi, patterns akan di-*filter* agar di *tree* tidak *double jikalau* memang ada *keyword* di *database* yang persis sama. Lalu dibuatlah *tree* oleh *function*

constructTree(). Setelah itu akan dilakukan pengecekan, jika karakter yang diketik tidak sesuai dengan karakter di dalam *tree*, akan dilakukan proses *failure function* yang mana akan mengecek ulang karakter dari *root tree* ke *state* lain. Jika sesuai, maka akan dimasukan ke dalam *variable* baru & dilakukan persamaan dengan *keyword* di dalam *database* yang telah diambil tadi. Jika sesuai, akan di-*return* dan di-*block* oleh aplikasi.

E. Sequence Diagram View All Keyword

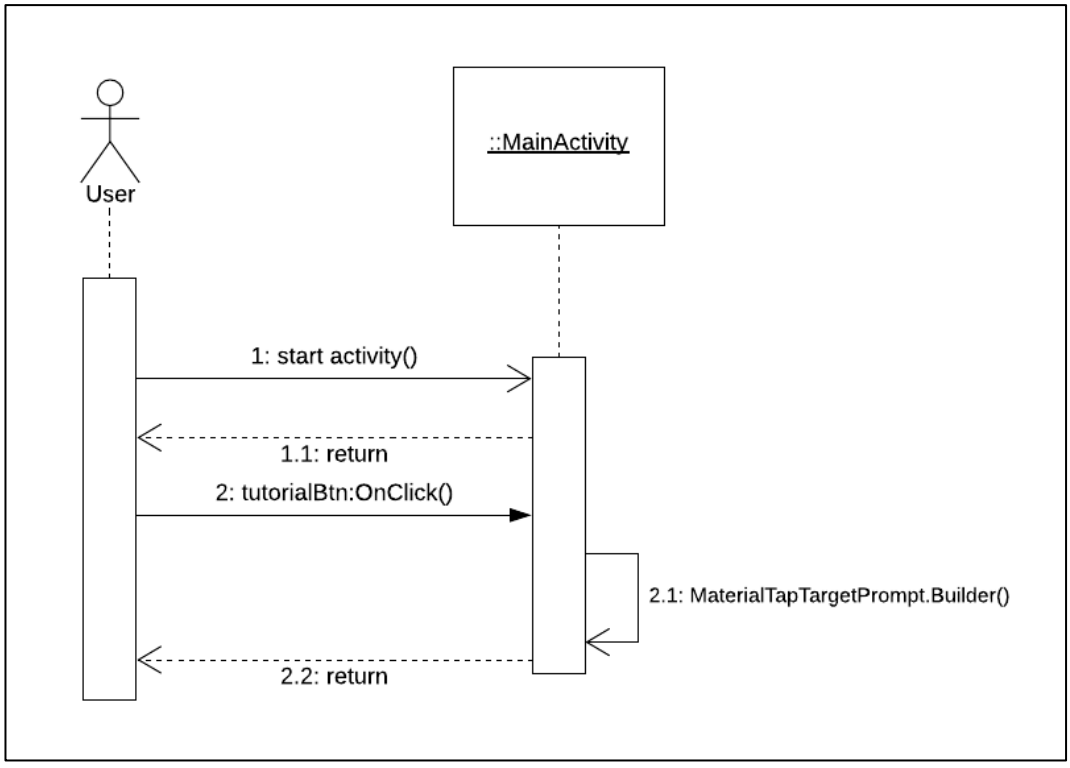
*Sequence diagram view all keyword* dapat dilihat pada Gambar 3.16. Untuk melakukan *view all keyword*, aplikasi pertama kali akan membuat *recycler view & cursor* untuk mengambil data *keyword* dari *database*. Lalu seluruh data *keyword* diambil, data tersebut akan di-*set* ke dalam *recycler view*. Setelah itu *recycler view* tersebut akan ditampilkan kepada *user*.




Gambar 3.16 *Sequence Diagram View All Keyword*

F. Sequence Diagram Tutorial

*Sequence diagram tutorial* dapat dilihat pada Gambar 3.17. Untuk melakukan *tutorial*, pertama kali *user* harus menekan *tutorial button* di *MainActivity*. Setelah itu aplikasi akan menjalankan *function* materialTapTargetPrompt.Builder() yang di mana merupakan *library*. Alur penampilan *material tap target prompt* dimulai dari *material tap target prompt untuk button add keyword*, setelah itu jika *button* atau area diluar *button* ditekan maka akan lanjut ke *material tap target prompt button list keyword*. Alur akan berjalan seperti itu sampai user menekan *material tap target prompt* untuk *button setting*. Setelah *material tap target prompt setting ditekan*, lalu *tutorial* pun akan selesai.

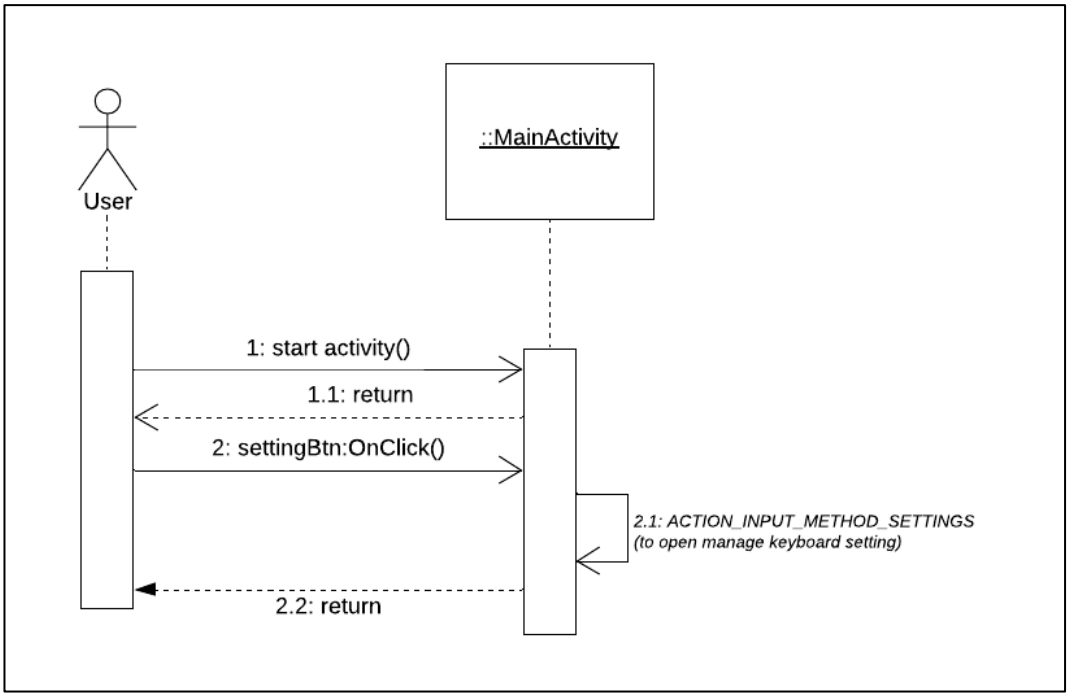



Gambar 3.17 *Sequence Diagram Tutorial*

G. Sequence Diagram Open Setting

*Sequence diagram open setting* dapat dilihat pada Gambar 3.18. Untuk membuka *setting*, pertama kali *user* harus menekan *open setting button* di

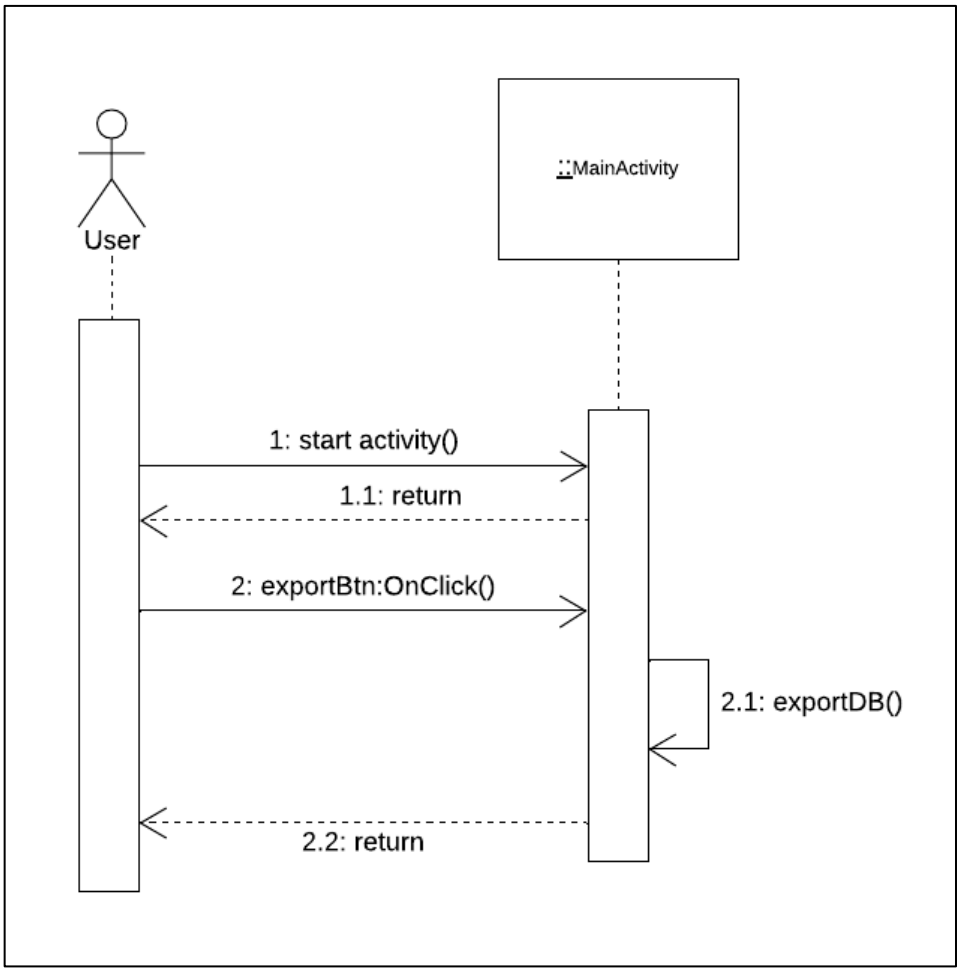
*MainActivity*. Setelah itu aplikasi akan langsung menampilkan *setting area* di *OS handphone* masing-masing di mana *user* dapat meng-*enable-kan* NFKeyword.




Gambar 3.18 *Sequence Diagram Open Setting*

H. Sequence Diagram Export Keyword

*Sequence diagram export keyword* dapat dilihat pada Gambar 3.19. Untuk melakukan *export keyword*, pertama kali *user* harus menekan *export button*. Setelah itu aplikasi akan langsung menjalankan *function* exportDB(). Di *function* ini terjadi aktifitas di dalam aplikasi yang akan mengambil *export directory* yang di mana jika belum dibuat sebelumnya, akan dibuat terlebih dahulu. Setelah itu aplikasi akan membuat *file* bernama *keyword.csv*, *cursor, dan CSVWriter* yang merupakan *library*. Lalu aplikasi akan mengambil dan menyimpan seluruh data yang ada di *database* dengan *cursor*. Setelah itu data satu-persatu akan di-*write* di *CSVWriter* tadi sampai data selesai, jika sudah selesai aplikasi akan langsung memberikan *feedback* kepada *user* bahwa proses *export* telah selesai dilakukan.

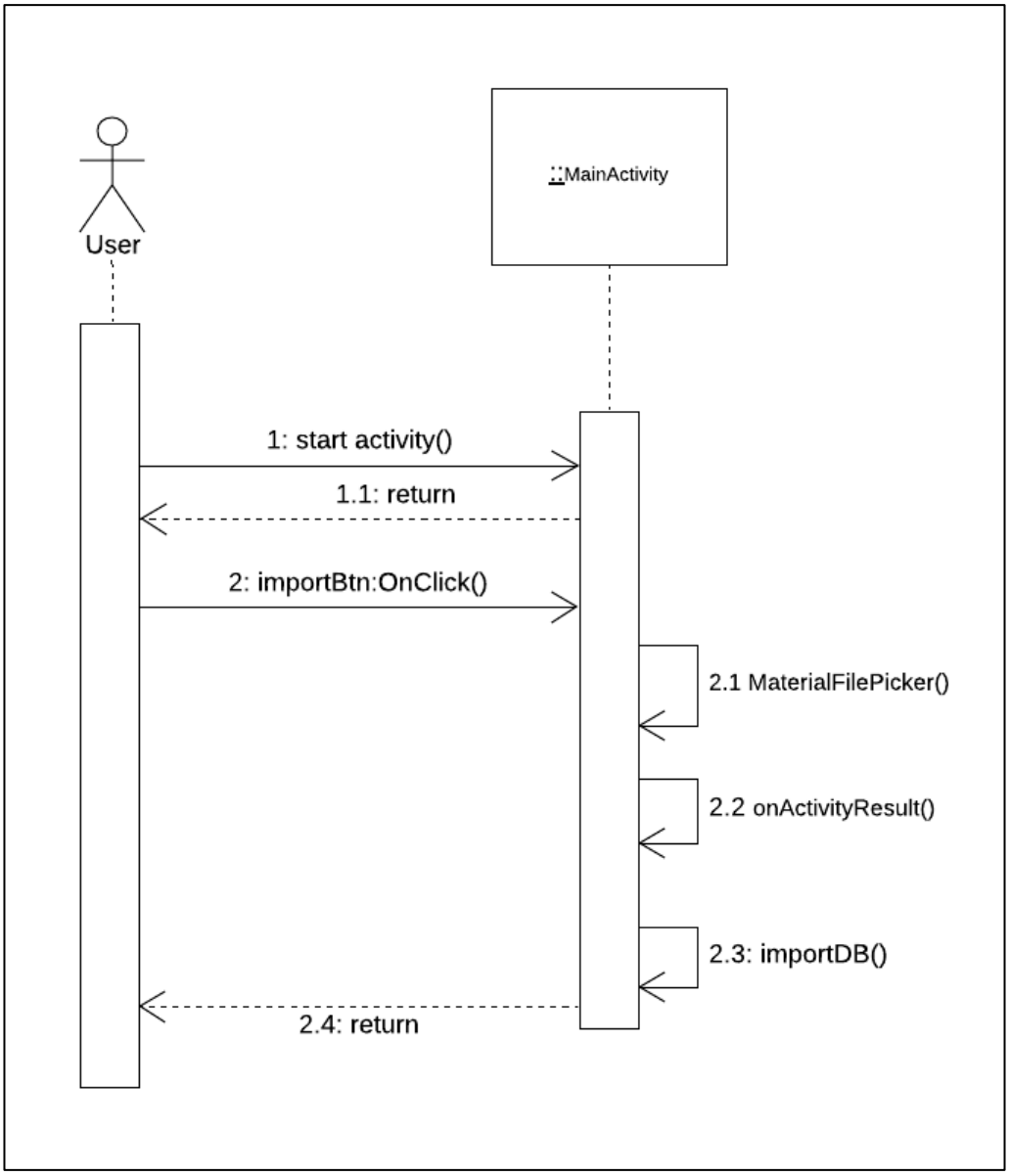



Gambar 3.19 *Sequence Diagram Export Keyword*

I. Sequence Diagram Import Keyword

*Sequence diagram import keyword* dapat dilihat pada Gambar 3.20. Untuk melakukan *import keyword*, pertama kali *user* harus menekan *import button* di *MainActivity*. Setelah itu aplikasi akan menjalankan function MaterialFilePicker() untuk membuka *internal storage*, *function* ini adalah *function* yang dijalankan dengan *library internal storage picker*. *File* selanjutnya akan dipilih oleh *user* dan divalidasi apakah merupakan *file* yang bernama *keyword.csv* atau bukan, proses ini ada di dalam *function* onActivityResult(). Jika iya, aplikasi akan mengambil *path* dari *file* tersebut dan menjalankan *function* importDB(). Di dalam proses importDB() aplikasi akan membuka & membaca seluruh isi *file*. Setelah itu isi *file*

akan di-*write* ke dalam *database*. Jika sudah selesai, aplikasi akan memberikan *feedback* pertanda bahwa *import* telah selesai dilakukan.

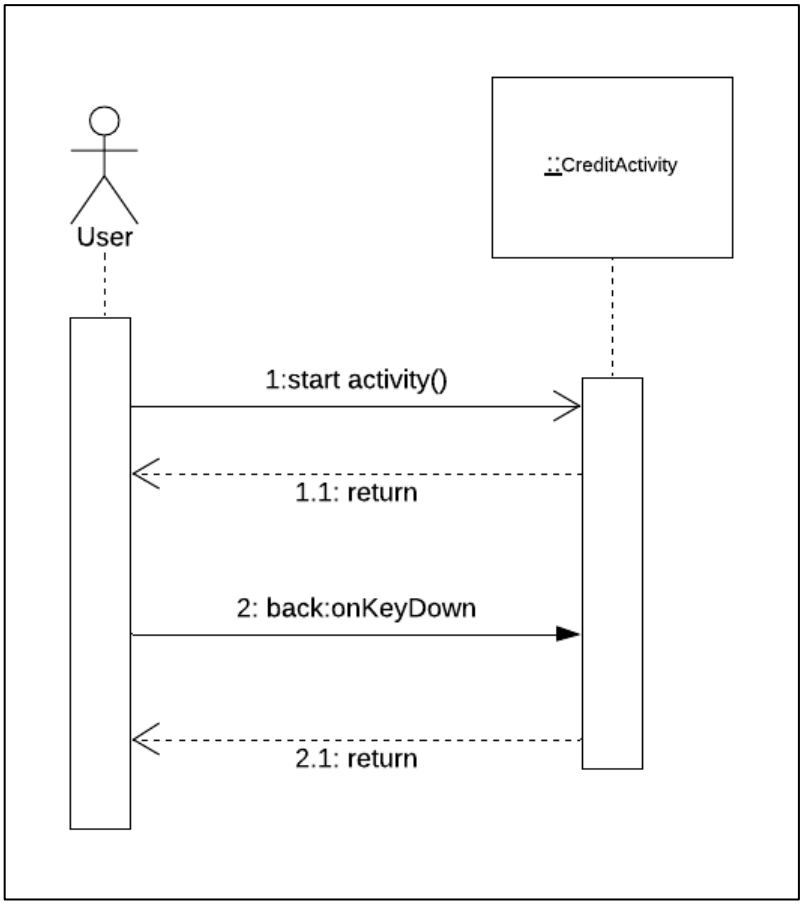



Gambar 3.20 *Sequence Diagram Import Keyword*

J. Sequence Diagram View Credits

*Sequence diagram view credits* dapat dilihat pada Gambar 3.21. Untuk melihat *credits*, pertama kali *user* harus menekan tombol *credist button* di MainActivity. Setelah itu akan langsung ditampilkan halaman *credits* kepada *user*. Jika sudah

selesai melihat halaman *credist, user* dapat kembali ke halaman utama dengan menekan tombol *back* pada *smartphone* masing-masing.

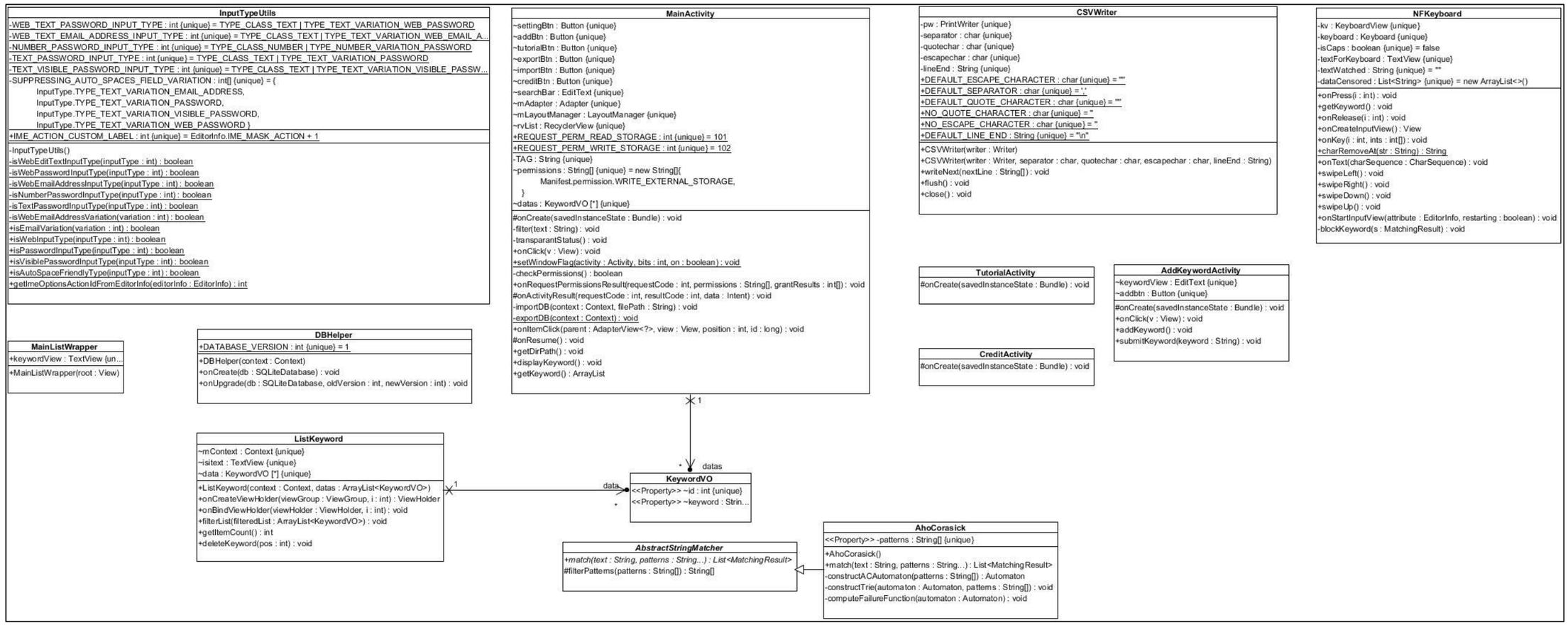



Gambar 3.21 *Sequence Diagram View Credits*

### 3.2.4 Class Diagram

*Class diagram* untuk aplikasi ini dapat dilihat pada Gambar 3.22. Terdapat beberapa *class* yang dibuat pada aplikasi ini. *Class* MainActivity, *Class* AddKeywordActivity, *Class* CreditsActivity, *Class* HelpActivity, digunakan untuk interaksi *user* pada aplikasi. *Class* AhoCorasick, *Class* AbstractStringMatcher digunakan untuk memroses data *keyword* yang akan di-*block*. *Class* CSVWriter yang merupakan *library* digunakan untuk *export keyword*. *Class* ListKeyword sebagai *adapter* dan ditampilkan ke aplikasi melalui *recyclerView*. *Class* DBHelper digunakan untuk memroses *table* pertama kali aplikasi di-*install* dan

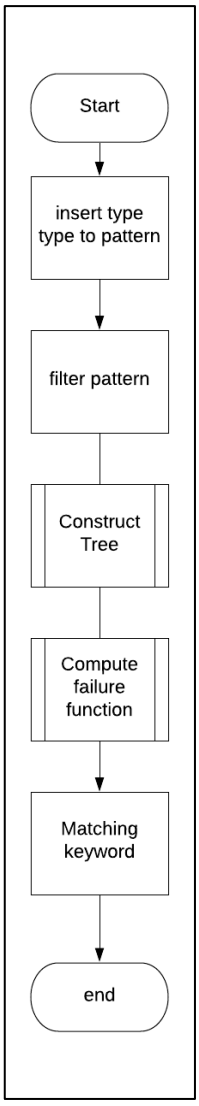
perubahan data yang berhubungan dengan skema *database*. *Class* KeywordVO menampung data *keyword*. *Class* NFKeyboard, *dan Class* InputTypeUtils digunakan untuk membuat *keyboard* aplikasi bawaan dan memproses data-data *keyword*.




Gambar 3.22 *Class Diagram*

### 3.2.5 Flowchart Aho-Corasick

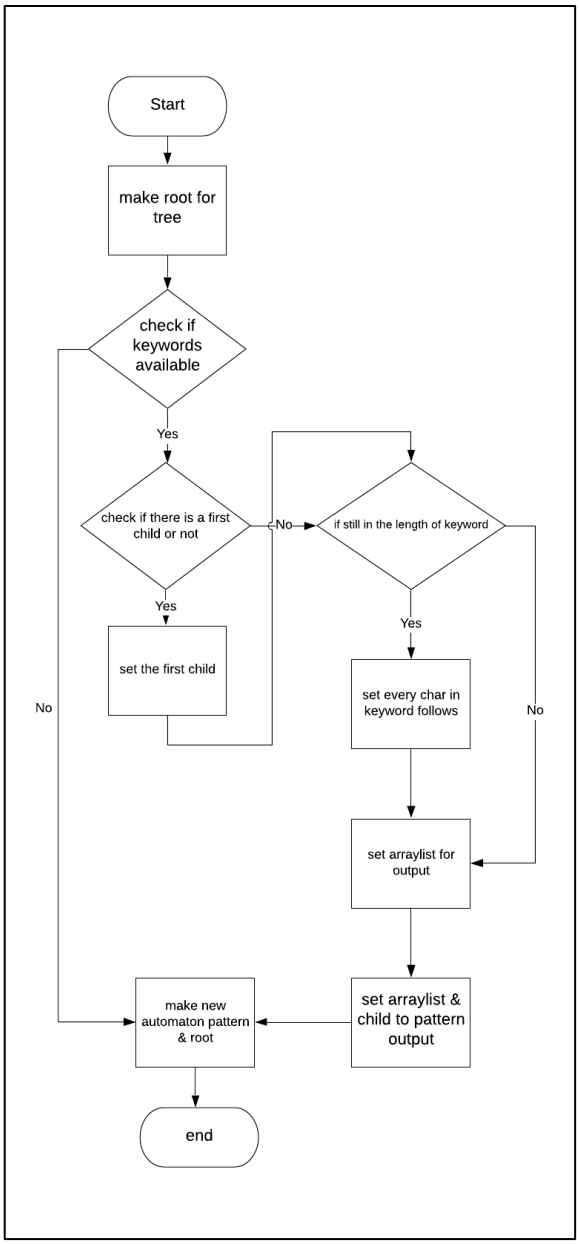
Gambar 3.23 menunjukan *flowchart* untuk algoritma Aho-corasick. Proses ini diawali dengan memasukan *keyword* yang ada di *database* ke dalam *pattern*.



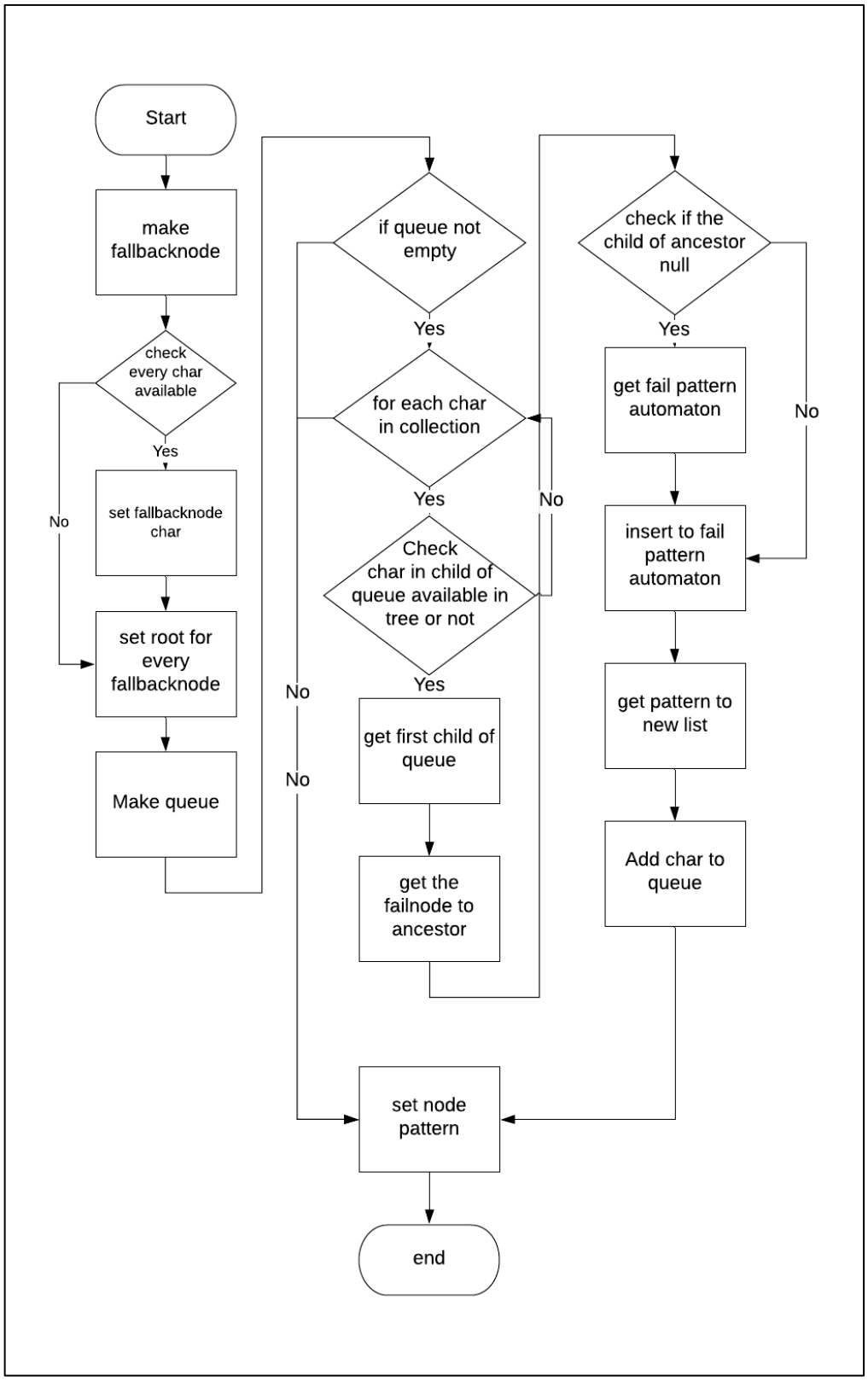

Gambar 3.23 *Flowchart* Aho-corasick

Lalu *pattern* tersebut akan di-*filter* sehingga jika ada *keyword* yang sama persis, tidak akan berulang pada saat pembuatan *tree*. Setelah di-filter, proses selanjutnya adalah membuat *tree*. *Tree* ini terbentuk dari semua *keyword* yang telah dimasukan *user* ke dalam *database*. Setelah *tree* terbentuk, lalu akan dilakukan proses *failure*

*function* yang di mana berfungsi untuk menentukan jika tidak ada *node* di dalam *tree* akan langsung dikembalikan kepada *root node* tersebut. Lalu proses berikutnya adalah *matching keyword*, untuk menentukan apakah sama antara karakter per kalimat yang *user* ketik dengan *keyword* yang sudah terbentuk di *tree*. Jika sudah di-*match*, hasilnya akan di-*return*.




Gambar 3.24 *Flowchart Construct Tree*




Gambar 3.25 *Flowchart Failure Function*

### 3.2.6 SQLite Database

*Database* yang digunakan dalam aplikasi ini adalah SQLite. Pada *database* tersebut terdapat 1 tabel yaitu tb_keyword. Adapun nama tabel, fungsi serta deskripsi tabel adalah sebagai berikut.

1. Nama Tabel : tb_keyword

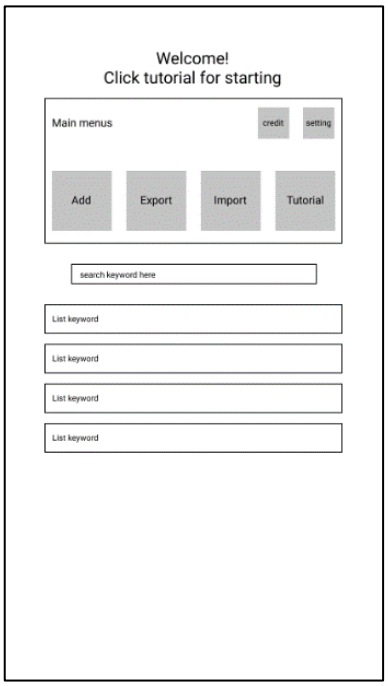
   Fungsi : Menyimpan nama dan id dari *keyword* yang *user* masukkan. Deskripsi tabel ditunjukkan pada Tabel 3.1.

Tabel 3.1 Tabel tb_keyword

| Nama Field | Tipe Data | Keterangan |
|---|---|---|
| id | int | Primary key |
| keyword | varchar | Keterangan kata pornografi |

### 3.2.7 Rancangan Tampilan Antarmuka

Perancangan tampilan antarmuka dilakukan dengan membuat sketsa yang menggambarkan seluruh tampilan aplikasi. Berikut rancangan tampilan antarcmuka pada aplikasi.

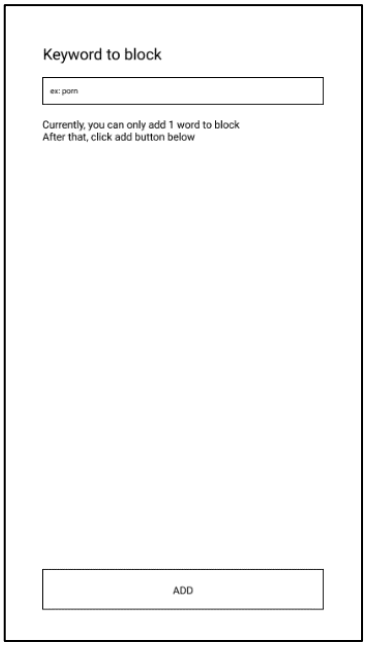



Gambar 3.26 Halaman Utama

Gambar 3.26 menunjukkan halaman utama dari aplikasi NFKeyboard. Pada halaman tersebut terdapat beberapa *button* pada bagian kanan atas. *Button* tersebut adalah *add, export, import, tutorial, credits,* dan *setting*. Pada bagian tengah terdapat *search bar* di mana *user* dapat langsung mengetik untuk mencari *keyword* yang ada. Di bawah *search bar*, terdapat *list keyword* yang nantinya akan dimunculkan setelah *user* meng-*add* atau meng-*import keyword*.

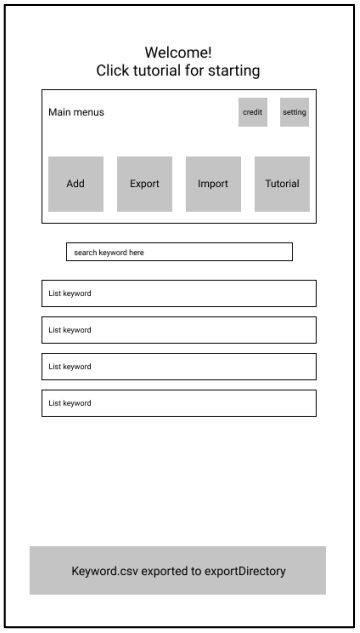
Gambar 3.27 menunjukkan halaman *add* dari aplikasi NFKeyboard. Pada halaman tersebut terdapat satu *button* pada bagian bawah dan satu *edit text* yang di mana merupakan tempat *user* untuk mengetik *keyword* untuk di-*add*. *Button add*

harus ditekan pada saat *user* telah selesai meng-*input keyword* pada *edit text* yang telah disediakan.




Gambar 3.27 Halaman *Add Keyword*

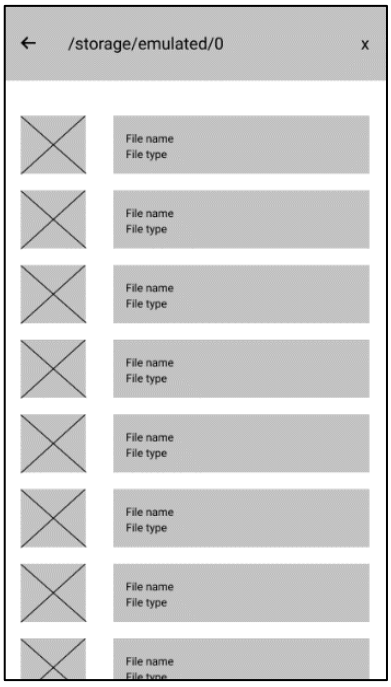
Gambar 3.28 menunjukkan keadaan setelah *button export* ditekan.




Gambar 3.28 *Export Keyword*

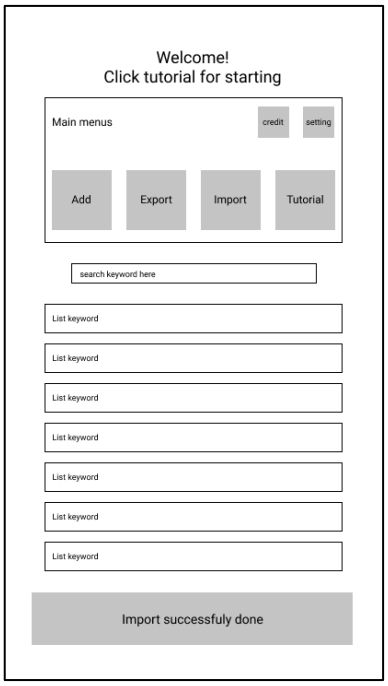
Aplikasi akan langsung memroses data. Setelah data diproses lalu aplikasi akan memberikan tanda bahwa *export* berhasil dilakukan dan disimpan ke dalam *directory smartphone*.

Gambar 3.29 menunjukkan keadaan setelah *button import* ditekan. Pertama-tama aplikasi akan membuka internal directory yang merupakan library dari *internal storage picker*. Setelah *internal directory* ditampilkan, *user* diharuskan untuk memilih *file* yang akan di-import. *File* yang di-*import* adalah *file* yang bernama keyword.csv.



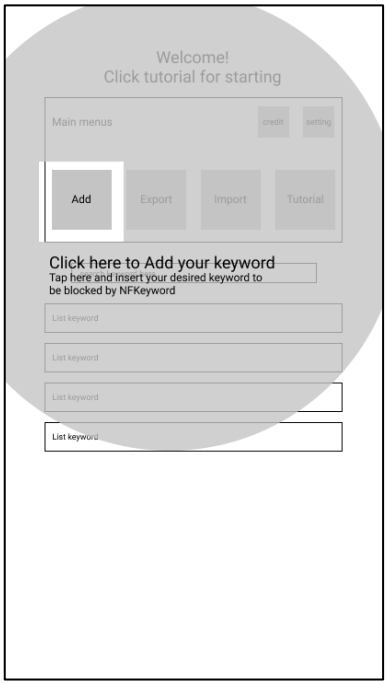

Gambar 3.29 Pilih file di *internal storage*

Gambar 3.30 menunjukkan keadaan setelah *file* telah dipilih *user* untuk di-*import*. Jika *file* telah divalidasi dan benar merupakan keyword.csv, aplikasi akan langsung memberikan tanda bahwa *import* berhasil dilakukan. *List keyword* pun akan bertambah.




Gambar 3.30 *Import keyword* berhasil

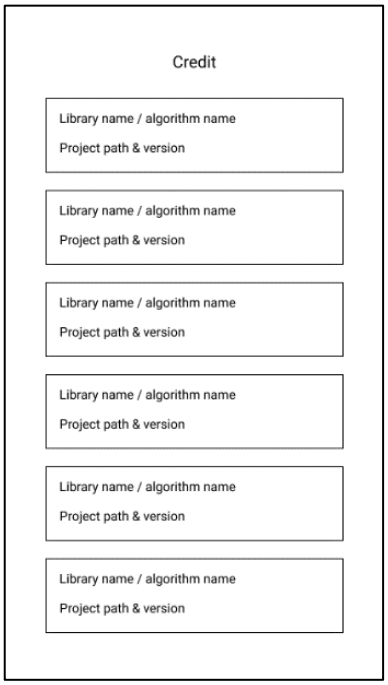
Gambar 3.31 menunjukkan keadaan jika *button tutorial* ditekan. Aplikasi akan langsung memberikan tampilan yang *focus* pada area di mana *tutorial* ditujukan.




Gambar 3.31 *Tutorial*

*Tutorial* akan lanjut ditampilkan setelah *user* menekan pada area sekitar. *Tutorial* akan berhenti jika telah mencapai *tutorial* untuk *setting*.

Gambar 3.32 menunjukkan halaman *view credits*. Aplikasi akan langsung menampilkan *library* dan algoritma apa saja yang dipakai dalam aplikasi ini. Pada halaman ini terdiri dari beberapa *card* yang memiliki bagian *header* berupa nama dan isi yang berupa *path* dari *project*.




Gambar 3.32 *View Credits*